Beyond Mechanism Design

David H. Wolpert[1] and Kagan Tumer[1]

NASA Ames Research Center, Moffett Field, CA, 94035, USA

Abstract. The field of mechanism design is concerned with setting (incentives superimposed on) the utility functions of a group of players so as to induce desirable joint behavior of those players. It arose in the context of traditional equilibrium game theory applied to games involving human players. This has led it to have many implicit restrictions, which strongly limits its scope. In particular, it ignores many issues that are crucial for systems that are large (and therefore far off-equilibrium in general) and/or composed of non-human players (e.g., computer-based agents). This also means it has concentrated on issues that are often irrelevant in those broader domains (e.g., incentive compatibility). This paper illustrates these shortcomings by reviewing some of the recent theoretical work on the design of collectives, a body of work that constitutes a substantial broadening of mechanism design. It then presents computer experiments based on a recently suggested nanotechnology testbed that demonstrates the power of that extended version of mechanism design.

1 MOTIVATION AND BACKGROUND

1.1 Collectives

This paper concerns **collectives**, which are defined as any system having the following two characteristics: First, the system must contain one or more agents each of which we view as trying to maximize an associated **private utility**. Second, the system must have an associated **world utility** function that rates the possible behaviors of that overall system [29,37–40].

While games are obviously collectives, the reverse need not be so, in the sense that what game underlies a particular collective may not be known. In practice collectives are often very large, distributed, and support little if any centralized communication and control, although those characteristics are not part of their formal definition. Collectives can be found in nature, as both organic and inorganic systems. They can also be artificial, e.g., computational.

We do not insist that the agents in a collective really are "trying" to maximize their private utilities, in some teleological sense. We only require only that they can be *viewed* that way. This allows us to circumvent the fraught exercise of formulating a definition of what an arbitrary component of some physical system is "trying to do". This is illustrated with a naturally occurring example of a collective involving gene expression in Eukaryotic cells [17]. Gene expression — the process of "reading" a gene into an associated protein according to the genetic code — requires numerous distinct

cellular machines. Each machine carries out a separate step in the expression pathway, with extensive coupling among the steps in the pathway. It is thought that the coupling optimizes the fidelity of the entire expression pathway. This allows us to view that expression pathway as a collective with large values of both the world utility and the private utilities. We do this by identifying the world utility with expression fidelity, the individual agents with the separate cellular machines, and the private utilities of those agents with quality of their performance at the associated roles they play in the entire pathway.

With the advent of ubiquitous cheap computing in the near future, the number of artificial control systems that are collectives should explode. Two obvious examples here are a user's constellation of multiple wearable computers, and "computational clouds" of computationally enabled household devices. If such distributed systems are not to be extremely brittle, then absent centralized communication and control, the individual components of the system will need to be both autonomous and adaptive. Almost by definition, this means that those components will be using statistical and machine learning techniques of some sort to modify their behavior to try to meet a goal, i.e., to maximize their private utility.[1] Moreover, in both of these examples, there is an obvious choice of world utility: the satisfaction level of the user(s) of the system.

As a final example, which will serve as the basis for our experiments reported below, consider search algorithms that try to find the value of a high-dimensional variable $\boldsymbol{z}$ for which a pre-specified function f has a large value. Examples of such algorithms are gradient ascent, simulated annealing, genetic algorithms, etc. Say we take the final value of f achieved by such an algorithm to be the "world utility" of the entire system's dynamic history. Assuming each individual component of $\boldsymbol{z}$ evolves with the "goal" of maximizing that final value of $f(\boldsymbol{z})$, we can view each such component as an agent, with private utility given by the final value of f. (Note that the private utility of an agent depends on variables not directly under the agent's control, in general.)

In this way any search algorithm can be viewed as a collective. However conventionally such algorithms use very "dumb" agents (e.g., semi-random agents rather than RL-based agents). They also don't consider possible modifications to the underlying system, e.g., to the choice of private utilities, that might result in a better value of final value of f. (The design problem of how best to set private utilities is discussed in the next section.) Constructing search algorithms that use techniques of this nature — intuitively, "agentizing" the individual variables of a search problem by providing them with adaptive intelligence — would provide a search algorithm that is immediately parallelizable. Owing to their use of "smart" variables, such algorithms

[1] When used for this purpose, such techniques are either explicitly or implicitly related to the field Reinforcement Learning (RL) [2,8,11,15,16,22,25,26,30].

might also lead to substantially better final values of f than conventional search algorithms.

1.2 The Design of Collectives and Mechanism Design

The "inverse problem" in the science of collectives is how one should initialize/update the precise configuration of the system — including in particular the private utilities of the individual agents — so that the ensuing behavior of the entire collective achieves large values of the provided world utility. Since in truly large systems detailed modeling of the system is usually impossible, it is crucial to try to solve this problem in a way that avoids such modeling. We need to solve it leveraging only the simple assumption that our agents' learning algorithms are individually fairly good at what they do.

This design problem is related to work in many other fields, including multi-agent systems (MAS's), computational economics, mechanism design, reinforcement learning, statistical mechanics, computational ecologies, (partially observable) Markov decision processes and game theory. However none of these fields is both applicable in large problems, and directly addresses the *general* inverse problem, rather than a special instance of it. (See [37] for a detailed discussion of the relationship between these fields, involving hundreds of references.)

For example, the subfield of game-theory known as mechanism design might, at first glance, appear to provide us techniques for solving the inverse problem. However mechanism design is almost exclusively concerned with collectives that are at (a suitable refinement of) Nash equilibrium [10,20,21]. That means that every agent is assumed to be performing *as well as is theoretically possible*, given the behavior of the rest of the system. In setting private utilities and the like on this basis, mechanism design ignores completely the issue of how to design the system so that each of the agents can achieve a good value of its private utility (given the behavior of the rest of the system). In particular it ignores all statistical issues related to how well the agents can be expected to perform for various candidate private utilities. Such issues become crucial as one moves to large systems, where each agent is implicitly confronted with a very high-dimensional RL task.

There are many other issues that arise in bounded rational situations that are not considered by mechanism design since they do not arise when there is full rationality. For example, it is often the case that if we "stabilize" the sequence of actions of some agent ρ, the other agents, being in a more predictable environment, are able to perform better. Conversely, such enforced stabilization of its actions will often hurt the performance of agent ρ. Mechanism design almost completely ignores the associated issues of how best to trade off the performance of one agent against that of other agents, or more generally of how best to trade off the degree of rationality of one agent against that of another agent. (Indeed, mechanism design does not

even possess a model-independent measure of "degree of rationality" that is both broadly applicable and appropriate for real-world domains.)

In addition to these problems, many of the techniques derived in mechanism design are inappropriate in numerous application domains, since those techniques are largely tailored to collectives of human beings. In particular, many of those techniques are tailored to the idiosyncracy of such collectives that their members have hidden variables whose values they "do not want to reveal". This idiosyncracy is often irrelevant in artificial systems where we get to design the agents *in toto*. Similarly, the need to view the private utility functions as fixed in stone, and therefore establish an elaborate mathematical structure involving "incentives" to allow us to affect the preference orderings of the agents, is usually unnecessary when we are designing the private utility functions of the agents from the ground up.

Conversely, its concentrating on humans has resulted in mechanism design's imposing strong restrictions on the allowed form of the private utilities and the world utility and communication structures among the agents, restrictions that may not hold in non-human systems. Indeed, if there were no such restrictions, then given the Nash equilibrium presumption of mechanism design, how best to set the private utilities would be a trivial problem: To have the maximum of world utility be a Nash equilibrium, simply set each such private utility to equal the world utility, in a so-called "team game" or "exact potential game" [9]. To have the analysis be non-trivial, restrictions like those that apply to the private utilities of human beings are needed.

Not only are the techniques of mechanism design not relevant in many domains, because those domains do not have the form assumed in mechanism design, but in addition there are many issues that loom large in such domains about which mechanism design is mute. For example, in computational domains, where the agents are computer programs each controlling a set of certain variables, we often have some freedom to change how the set of all variables being controlled is partitioned among the agents, and even change the number of such agents. Needless to say, with its focus on human agents, mechanism design has little advice to provide on such issues of how best to define the agents in the first place.

Perhaps the most striking illustration of the shortcoming of mechanism design is the fact that it does not allow for run-time adaptive redesign. For real-world bounded rational agents, the initial design of the system necessarily makes assumptions which invariably are at least partially at variance with reality. To address this, one must employ adaptive techniques (e.g., statistical estimation) on the running system to refine one's initial assumptions, and then modify the design accordingly. Yet almost all of mechanism design has no room for addressing such "macro-learning".

There is other previous work that does consider the inverse problem in its broadest sense, and even has each agent explicitly use RL techniques, so that no formal assumption is made in the associated theory that the system

is at Nash equilibrium. Despite this use of RL though, in general in that work the private utilities are set as in a team game. So again, there is no concern for how well the agents can discern how best to act to maximize their utilities. Unfortunately, as intimated above (and expounded below), ignoring this issue means that the approach scales extremely poorly to large problems. Intuitively, the difficulty is that each agent will have a hard time discerning the echo of its behavior on its private utility when the system is large if that private utility is the world utility; each agent has a horrible "signal-to-noise" problem in such a situation.[2]

Intuitively, in designing the private utilities of a collective we want them to be "aligned" with the world utility, in that modifications an agent might make that would increase its private utility also must increase world utility. Fortunately the equivalence class of such private utilities extends well beyond team-game utilities. In particular, it extends to include utilities that have far better "signal-to-noise" properties. By using those utilities one can get far better values of world utility than would otherwise be possible. The mathematical theory for how to generate such alternative private utilities is presented in the next section. The following section of this chapter then summarizes many experiments that demonstrate that by using those alternative private utilities one can improve performance by up to orders of magnitude, and that the gain in performance grows as the system gets larger.

2 The Mathematics of Designing Collectives

In this chapter attention is restricted to collectives in which the individual agents are pre-fixed, being the players in multi-stage non-cooperative games, with their moves at any single stage in no *a priori* way restricted by their moves at other times or by the moves of the other players. Some techniques for the design of the private utilities in such games are known as the "COllective INtelligence (COIN)" framework.[38] This section presents some of the mathematics necessary to understand that framework. It should be emphasized however that the full mathematics of how to design collectives extends significantly beyond what is needed to address such games.[3]

The restricted version of that full mathematics needed to present the COIN framework starts with an arbitrary vector space Z whose elements ζ

[2] To help see this, consider the example of a collective provided by the human economy. A team game in that example would mean that every human gets US GDP as its reward signal, and tries to discern how best to act to maximize that reward signal. At the risk of understatement, this would provide the individual members of the economy with a difficult reinforcement learning task.

[3] That framework encompasses, for example, arbitrary dynamic redefinitions of the "players" (i.e., dynamic reassignments of how the various subsets of the variables comprising the collective across all space and time are assigned to players), as well as modification of the players' information sets (i.e., modification of inter-player communication). See [33].

give the joint move of all players in the collective in some stage. We wish to search for the ζ that maximizes the provided world utility $G(\zeta)$. In addition to G we are concerned with private utility functions $\{g_\eta\}$, one such function for each variable/player η. We use the notation $\hat{\eta}$ to refer to all players other than η.

We will need to have a way to "standardize" utility functions so that the numeric value they assign to a ζ only reflects their ranking of ζ relative to certain other elements of Z. We call such a standardization of some arbitrary utility U for player η the "**intelligence** for η at ζ with respect to U". Here we will use intelligences that are equivalent to percentiles:

$$\epsilon_{\eta,U}(\zeta) \equiv \int d\mu_{\zeta_{\hat{\eta}}}(\zeta')\Theta[U(\zeta) - U(\zeta')] \; , \tag{1}$$

where the Heaviside function Θ is defined to equal 1 when its argument is greater than or equal to 0, and to equal 0 otherwise, and where the subscript on the (normalized) measure $d\mu$ indicates it is restricted to ζ' sharing the same non-η components as ζ.[4] Intelligence value are always between 0 and 1.

Our uncertainty concerning the behavior of the system is reflected in a probability distribution over Z. Our ability to control the system consists of setting the value of some characteristic of the collective, e.g., setting the private utility functions of the players. Indicating that value of the **global coordinate** by s, our analysis revolves around the following **central equation** for $P(G \mid s)$, which follows from Bayes' theorem:

$$P(G \mid s) = \int d\boldsymbol{\epsilon}_G P(G \mid \boldsymbol{\epsilon}_G, s) \int d\boldsymbol{\epsilon}_g P(\boldsymbol{\epsilon}_G \mid \boldsymbol{\epsilon}_g, s) P(\boldsymbol{\epsilon}_g \mid s) \; , \tag{2}$$

where $\boldsymbol{\epsilon}_g \equiv (\epsilon_{\eta_1,g_{\eta_1}}(\zeta), \epsilon_{\eta_2,g_{\eta_2}}(\zeta), \cdots)$ is the vector of the intelligences of the players with respect to their associated private utility functions, and $\boldsymbol{\epsilon}_G \equiv (\epsilon_{\eta_1,G}(\zeta), \epsilon_{\eta_2,G}(\zeta), \cdots)$ is the vector of the intelligences of the players with respect to G.

Note that $\epsilon_{\eta,g_\eta}(\zeta) = 1$ means that player η is fully rational at ζ, in that its move maximizes the value of its utility, given the moves of the players. In other words, a point ζ where $\epsilon_{\eta,g_\eta}(\zeta) = 1$ for all players η is one that meets the definition of a game-theory Nash equilibrium.[5] On the other hand, a ζ at which all components of $\boldsymbol{\epsilon}_G = 1$ is a local maximum of G (or more precisely,

[4] The measure must reflect the type of system at hand, e.g., whether Z is countable or not, and if not, what coordinate system is being used. Other than that, any convenient choice of measure may be used and the theorems will still hold.

[5] See [10]. Note that consideration of points ζ at which *not* all intelligences equal 1 provides the basis for a model-independent formalization of bounded rationality game theory. This formalization contains variants of many of the theorems of conventional full-rationality game theory. See [32].

a critical point of the $G(\zeta)$ surface). So if we can get these two vectors to be identical, then if the agents do well enough at maximizing their private utilities we are assured we will be near a local maximum of G.

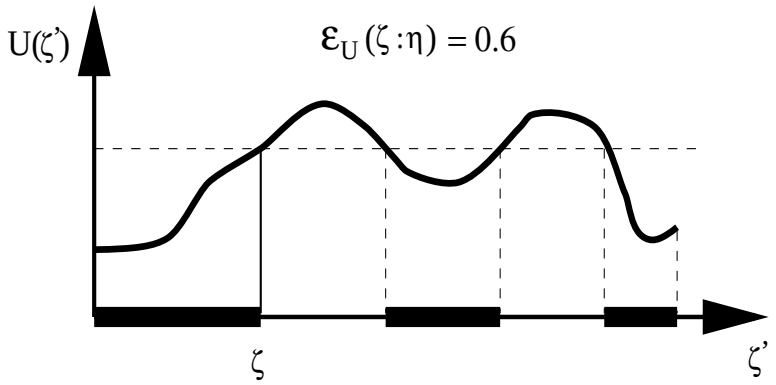

Fig. 1. Intelligence of agent η at state ζ for utility U is the actual joint move at hand. The x-axis shows agent η's alternative possible moves (all states ζ' having ζ's values for the moves of all players other than η.). The thick sections of the x-axis show the alternative moves that η could have made that would have given η a worse value of the utility U. The fraction of the full set of η's possible moves that lies in those thick sections (which is 0.6 in this example) is the intelligence of agent η at ζ for utility U, denoted by $\epsilon_{\eta,U}(\zeta)$.

To formalize this, consider our decomposition of $P(G \mid s)$. If we can choose s so that the third conditional probability in the integrand is peaked around vectors $\boldsymbol{\epsilon}_g$ all of whose components are close to 1, then we have likely induced large (private utility function) intelligences. If we can also have the second term be peaked about $\boldsymbol{\epsilon}_G$ equal to $\boldsymbol{\epsilon}_g$, then $\boldsymbol{\epsilon}_G$ will also be large. Finally, if the first term in the integrand is peaked about high G when $\boldsymbol{\epsilon}_G$ is large, then our choice of s will likely result in high G, as desired.

Intuitively, the requirement that private utility functions have high "signal-to-noise" arises in the third term. It is in the second term that the requirement that the private utility functions be "aligned with G" arises. In this chapter we concentrate on these two terms, and show how to simultaneously set them to have the desired form. [6]

[6] Search algorithms that do not involve game theory (e.g., simulated annealing) can be viewed as addressing how to have term 1 have the desired form. They do this by trying to ensure that the particular local maximum they find of the function they are searching has a high value of that function. This is the essence of why such algorithms "trade off exploration and exploitation". One can combine such term-1-based techniques with the techniques presented in this paper that concentrate on terms 2 and 3. Intuitively, this amounts to "wrapping" a

Details of the stochastic environment in which the collective operates, together with details of the learning algorithms of the players, are all bundled into the distribution $P(\zeta)$ which underlies the distributions appearing in Equation 2. Note though that *independent of these considerations*, our desired form for the second term in Equation 2 is assured if we have chosen private utilities such that ϵ_g equals ϵ_G exactly for all ζ. Such a system is said to be **factored**. In game-theory parlance, the Nash equilibria of a factored collective are local maxima of G. In addition to this desirable equilibrium behavior, factored collectives also automatically provide appropriate off-equilibrium incentives to the players (an issue rarely considered in the game theory / mechanism design literature).

As a trivial example, any "team game" in which all the private utility functions equal G is factored [9,18]. However team games often have very poor forms for term 3 in Equation 2, forms which get progressively worse as the size of the collective grows. This is because for such private utility functions each player η will usually confront a very poor "signal-to-noise" ratio in trying to discern how its actions affect its utility $g_\eta = G$, since so many other player's actions also affect G and therefore dilute η's effect on its own private utility function.

We now focus on algorithms based on private utility functions $\{g_\eta\}$ that optimize the signal/noise ratio reflected in the third term, subject to the requirement that the system be factored. To understand how these algorithms work, say we are given an arbitrary function $f(\zeta_\eta)$ over player η's moves, two such moves ${\zeta_\eta}^1$ and ${\zeta_\eta}^2$, a utility U, a value s of the global coordinate, and a move by all players other than η, $\zeta_{\hat{\eta}}$. Define the associated **learnability** by

$$\Lambda_f(U; \zeta_{\hat{\eta}}, s, {\zeta_\eta}^1, {\zeta_\eta}^2) \equiv \sqrt{\frac{[E(U; \zeta_{\hat{\eta}}, {\zeta_\eta}^1) - E(U; \zeta_{\hat{\eta}}, {\zeta_\eta}^2)]^2}{\int d\zeta_\eta [f(\zeta_\eta) Var(U; \zeta_{\hat{\eta}}, \zeta_\eta)]}} \; . \tag{3}$$

The expectation values in the numerator are formed by averaging over the training set of the learning algorithm used by agent η, n_η. Those two averages are evaluated according to the two distributions $P(U|n_\eta)P(n_\eta|\zeta_{\hat{\eta}}, {\zeta_\eta}^1)$ and $P(U|n_\eta)P(n_\eta|\zeta_{\hat{\eta}}, {\zeta_\eta}^2)$, respectively. (That is the meaning of the semicolon notation.) Similarly the variance being averaged in the denominator is over n_η according to the distribution $P(U|n_\eta)P(n_\eta|\zeta_{\hat{\eta}}, \zeta_\eta)$.

The denominator in Equation 3 reflects how sensitive $U(\zeta)$ is to changing $\zeta_{\hat{\eta}}$. In contrast, the numerator reflects how sensitive $U(\zeta)$ is to changing ζ_η. So the greater the learnability of a private utility function g_η, the more $g_\eta(\zeta)$ depends only on the move of player η, i.e., the better the associated

system using the private utilities derived below in an outer loop that trades off exploration and exploitation. The resultant hybrid "Computational Corporation" (CoCo), algorithm, addressing all three terms, outperforms simulated annealing by over two orders of magnitude[34]. Due to its adding in concern for term 1, CoCo also typically outperforms the techniques presented in this paper; it is only for reasons of space that we do not explore it here.

signal-to-noise ratio for η. Intuitively then, so long as it does not come at the expense of decreasing the signal, increasing the signal-to-noise ratio specified in the learnability will make it easier for η to achieve a large value of its intelligence. This can be established formally: if appropriately scaled, g'_η will result in better expected intelligence for agent η than will g_η whenever $\Lambda_f(g'_\eta; \zeta_{\hat{\eta}}, s, {\zeta_\eta}^1, {\zeta_\eta}^2) > \Lambda_f(g_\eta; \zeta_{\hat{\eta}}, s, {\zeta_\eta}^1, {\zeta_\eta}^2)$ for all pairs of moves ${\zeta_\eta}^1, {\zeta_\eta}^2$[33]. [7]

It is possible to solve for the set of all private utilities that are factored with respect to a particular world utility. Unfortunately, in general it is not possible for a collective both to be factored and to have infinite learnability for all of its players. However consider **difference** utilities, which are of the form

$$U(\zeta) = \beta[G(\zeta) - \Gamma(\zeta_{\hat{\eta}})] \tag{4}$$

Any difference utility is factored [33]. In addition, under usually benign approximations, $\Lambda_f(U; \zeta_{\hat{\eta}}, s, {\zeta_\eta}^1, {\zeta_\eta}^2)$ is maximized over the of difference utilities for all pairs ${\zeta_\eta}^1, {\zeta_\eta}^2$ by choosing

$$\Gamma(\zeta_{\hat{\eta}}) = E(f(\zeta_\eta)G(\zeta) \mid \zeta_{\hat{\eta}}, s) \ , \tag{5}$$

up to an overall additive constant, where the expectation value is over ζ_η. We call the resultant difference utility the **Aristocrat** utility (AU), loosely reflecting the fact that it measures the difference between a player's actual action and the average action. If each player η uses an appropriately rescaled version of the associated AU as its private utility function, then we have ensured good form for both terms 2 and 3 in Equation 2.

Using AU in practice is sometimes difficult, due to the need to evaluate the expectation value. Fortunately there are other utility functions that, while being easier to evaluate than AU, still are both factored and possess superior learnability to the team game utility, $g_\eta = G$. One such private utility function is the **Wonderful Life** Utility (WLU). The WLU for player η is parameterized by a pre-fixed **clamping parameter** CL_η chosen from among η's possible moves:

$$WLU_\eta \equiv G(\zeta) - G(\zeta_{\hat{\eta}}, CL_\eta) \ . \tag{6}$$

WLU is factored no matter what the choice of clamping parameter. Furthermore, while not matching the high learnability of AU, WLU usually has far better learnability than does a team game, and therefore (when appropriately scaled) results in better expected intelligence [29,39,37,41].

[7] In many RL algorithms, changing the scale of the utility is exactly equivalent to changing a "temperature" parameter of the algorithm. Such temperatures have to usually be set via a search process. The result presented here establishes that so long as g'_η has higher learnability than does g_η, the expected intelligence of g'_η at the associated optimal temperature will be higher than that of g_η at its optimal temperature.

$$\begin{array}{c} \\ \eta_1 \\ \eta_2 \\ \eta_3 \\ \eta_4 \end{array} \begin{array}{c} \zeta \\ \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right] \end{array} \begin{array}{c} \\ \Longrightarrow \\ \text{Clamp } \eta_2 \\ \text{to "null"} \end{array} \begin{array}{c} (\zeta_{\neg\eta_2}, \mathbf{0}) \\ \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right] \end{array} \begin{array}{c} \\ \Longrightarrow \\ \text{Clamp } \eta_2 \\ \text{to "average"} \end{array} \begin{array}{c} (\zeta_{\neg\eta_2}, \boldsymbol{a}) \\ \left[\begin{array}{ccc} 1 & 0 & 0 \\ .33 & .33 & .33 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right] \end{array}$$

Fig. 2. This example shows the impact of the clamping operation on the joint state of a four-player system where each player has three possible moves, each such move represented by a three-dimensional unary vector. The first matrix represents the joint state of the system ζ where player 1 has selected action 1, player 2 has selected action 3, player 3 has selected action 1 and player 4 has selected move 2. The second matrix displays the effect of clamping player 2's action to the "null" vector (i.e., replacing ζ_{η_2} with $\mathbf{0}$). The third matrix shows the effect of instead clamping player 2's move to the "average" action vector $\boldsymbol{a} = \{.33, .33, .33\}$, which amounts to replacing that player's move with the "illegal" move of fractionally taking each possible move ($\zeta_{\eta_2} = \boldsymbol{a}$).

Figure 2 provides an example of clamping. As in that example, in many circumstances there is a particular choice of clamping parameter for player η that is a "null" move for that player, equivalent to removing that player from the system. (Hence the name of this private utility function — cf. the Frank Capra movie.) For such a clamping parameter assigning the associated WLU to η as its private utility function is closely related to the economics technique of "endogenizing a player's externalities", for example with the Groves mechanism [19,20,10].

However it is usually the case that using WLU with a clamping parameter that is as close as possible to the expected move defining AU results in far higher learnability than does clamping to the null move. Such a WLU is roughly akin to a mean-field approximation to AU.[8] For example, in Fig. 2, if the probabilities of player 2 making each of its possible moves was 1/3, then one would expect that a clamping parameter of $\boldsymbol{a}$ would be close to optimal. Accordingly, in practice use of such an alternative WLU derived as a "mean-field approximation" to AU almost always results in far better values of G than does the "endogenizing" WLU.

Intuitively, collectives having factored and highly learnable private utilities like AU can be viewed as akin to well-run human companies. G is the "bottom line" of the company, the players η are identified with the employees of that company, and the associated g_η given by the employees' performance-based compensation packages. For example, for a "factored company", each

[8] Formally, our approximation is exact only if the expected value of G equals G evaluated at the expected joint move (both expectations being conditioned on given moves by all players other than η). In general though, for relatively smooth G, we would expect such a mean-field approximation to AU, to give good results, even if the approximation does not hold exactly.

employee's compensation package contains incentives designed such that the better the bottom line of the corporation, the greater the employee's compensation. As an example, the CEO of a company wishing to have the private utilities of the employees be factored with G may give stock options to the employees. The net effect of this action is to ensure that what is good for the employee is also good for the company. In addition, if the compensation packages are "highly learnable", the employees will have a relatively easy time discerning the relationship between their behavior and their compensation. In such a case the employees will both have the incentive to help the company and be able to determine how best to do so. Note that in practice, providing stock options is usually more effective in small companies than in large ones. This makes perfect sense in terms of the formalism summarized above, since such options generally have higher learnability in small companies than they do in large companies, in which each employee has a hard time seeing how his/her moves affect the company's stock price.

3 Tests of the Mathematics

3.1 Previous experiments

As a test of the preceding mathematics, in some of our previous work we used the WLU for distributed control of network packet routing [39], achieving substantially better throughput than by using the best possible shortest-path-based system [39], even though that SPA-based system has information denied the agents in the WLU-based collective. In related work we have shown that use of the WLU automatically avoids the infamous Braess' paradox, in which adding new links can actually decrease throughput — a situation that readily ensnares SPA's [29,36]. In yet other work we have applied the WLU to the problem of controlling communication across a constellation of satellites so as minimize the importance-weighted loss of scientific data flowing across that constellation [35]. We have also successfully applied COIN techniques to the problem of coordinativing a set of autonomous rovers so as to maximize the importance-weighted value of a set of locations they visit [28].

In addition we have explored COIN-based techniques on variants of congestion games [38,40,41], in particular of a more challenging variant of Arthur's El Farol bar attendance problem [1], sometimes also known as the "minority game" [7]. In this work we showed that use of the WLU can result in performance *orders of magnitude* superior to that of team game utilities.

3.2 The faulty devices choice problem

In addition to the examples of the previous subsection, we have successfully applied the COIN techniques to problems that are explicitly cast as search.

These include setting the states of the spins in a spin glass to minimize energy; the conventional bin-packing problem of computer science, and a model of human agents connected in a small-world network who have to synchronize their purchase decisions.

Here we explore the use of these COIN techniques for a new problem recently proposed by Challet and Johnson [6]. This problem is an abstraction of a problem expected to loom large in nanotechnology: given a collection of faulty devices, how to choose the subset of those devices that, when combined with each other, gives optimal performance. The canonical version of this problem arises when the devices are all observational devices producing a single real number whose value is a (fixed) distortion of the true value that would be read by a perfect device. The distortions are assigned to each device by random sampling of a fixed Gaussian distribution. The problem is to choose the subset of a fixed collection of such devices to have the average (over the members of the subset) distortion as close to zero as possible.

Formally, the problem is to minimize

$$\epsilon \equiv \frac{|\sum_{j=1}^{N} n_j a_j|}{\sum_{k=1}^{N} n_k} ,$$

where $n_j \in \{0, 1\}$ is whether device j is or is not selected, and there are N devices in the collection, having associated distortions $\{a_j\}$. We identify ϵ with the world utility, G. There are N individual agents, each setting one of the n_j. The goal is to give those agents private utilities so that, as they learn to maximize their private utilities, the maximizer of G is found.

3.3 Experiment details

Since we wished to concentrate on the effects of the utilities rather than on the RL algorithms that use them, we used (very) simple RL algorithms.[9] We would expect that even marginally more sophisticated RL algorithms would give better performance.

In the algorithm used in this study, each player η had a 2-dimensional vector giving its estimates of the utility it would receive for taking each possible move. At the beginning of each week, each η picked whether to set its action n_η to one or zero, using a Boltzmann distribution over the two components of η's estimated utilities vector. For simplicity, given how short our runs were, temperature did not decay in time. However to reflect the fact that each player operated in a non-stationary environment, utility estimates were formed using exponentially aged data: for any time step t, the utility

[9] Indeed, to use algorithms any simpler than the one we used, algorithms so patently deficient that they have never even been considered in the RL community — like the algorithms used in much of the bar problem literature — would seriously interfere with our ability to interpret our experiments.

estimate η uses for setting either of the two actions n_η was a weighted average of all the utility values it had received at previous times t' that it chose that action, with the weights in the average given by an exponential of the values $t - t'$. To form the players' initial training set of action-(utility value) pairs, we had an initial period in which all actions by all players were chosen uniformly randomly, with no learning. It was after this initial period that the agents began choosing their actions according to the associated Boltzmann distributions.

Figures 3 and 4 show the convergence properties of different algorithms in systems with 100 and 1000 agents, respectively. The results reported are based on 20 different $\{a_j\}$ configurations, each performed 50 times (i.e., each point on the figure is the average of $20 \times 50 = 1000$ runs). G, AU, and WLU show the performance of agents using reinforcement learners with those reinforcement signals provided by G (team game), Aristocrat Utility and Wonderful Life Utility respectively. S shows the performance of greedy search where new n_j's are generated at each step and selected if the solution is better than the current best solution. Because the runs are only 200 timesteps long, algorithms such as simulated annealing do not outperform simple search: there is simply no time for an annealing schedule.

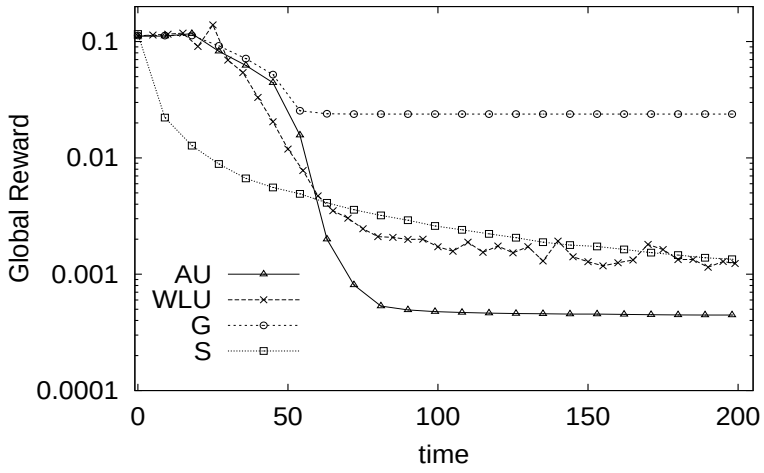

Fig. 3. Faulty Devices Choice Problem, N=100.

For all learning algorithms, the first 20 time steps constitute the "initial training set gathering" phase where each agent takes random actions and stores the resulting utility values (hence all learning algorithms perform the same during that time). Starting at $t = 20$, with each consecutive timestep a fixed fraction of the agents still choosing their actions randomly switch to